LaTex (tikz) 转换为图像

发布 *1.0.0*

高斯羽 博士 (Dr. Gāo, Sī Yǔ)

2019 年 10 月 24 日

目录

此项目是一个关于把 LaTex 文档直接转换为各种图像的教程（在编译 TEX 文件时，同时生成单独的图片）。此教程主要关注如何把 tikz 生成的，内嵌于 LaTex 生成的 PDF 文件中的图像转换为各种格式的单独图片。

此项目会讨论到的图片格式如下

- SVG （矢量图）
- PNG（位图）
- EMF （Windows 系统上的矢量图）
- EPS （印刷常用格式）

此项目的在于提供基于 Windows 系统的教程和例子。作者相信 Linux 用户有能力独自解决这个问题。

此教程会提供软件安装和配置指南，并会结合例子进行讲解。

此教程认为用户已经对 LaTex 有一定的理解，因而不会对 LaTex 中之各种进行详解。

本教程将会详尽讲解流程。若只想快速使用而不在乎原理，可先阅读软件的安装和配置 然后按照极简教程 中之步骤执行即可。

CHAPTER 1

系统和软件

此项目会用到如下的系统和软件，请先保证你已安装了它们。

- Windows 7 或 Windows 10
- texlive（2019 年的发布，免费）
- LaTex 的 `standalone` 包（`texlive` 自带）
- `pdftocairo` （`texlive` 自带，用于 PNG 转换）
- pdf2svg（Windows 编译版，免费。用于 SVG 转换）
- inksacpe（0.92.4，免费。用于 EMF 和 EPS 的转换）
- ghostscript (9.50，免费。用于 PDF 文件的分页)

1.1 转换软件的使用理由

此项目选择的转换软件主要基于以下理由。

- 此项目坚持所有使用的转换软件必须为免费
- 转换结果必须是一页 PDF 一张图
- 当把 PDF 转换为矢量图时，必须为真正的矢量图，而不是包裹在矢量格式中的位图
  - 因此，此项目不使用 ImageMagick，因其在转换矢量图时经常栅格化
- 当转为为矢量图时，字体应该嵌入而不是栅格化
- 转换命令应尽可能简单
- 使用的软件尽可能少以降低依赖性

注解： 尽管 inkscape 也可以进行 PNG 和 SVG 的转换，但 pdftocairo 和 pdf2svg 自带了多页到单页的功能，使用便利，而且安装也简易，故而用此二软件分别进行 PNG 和 SVG 的转换而不使用 inkscape 。若使用 inkscape ，则需要先调用 ghostscript 对 PDF 进行分页，然后再转换。

CHAPTER 2

软件的安装和配置

软件的下载链接可在*系统和软件* 获得。若链接失效，请自行搜索。

- `texlive` 的安装过程比较长，请耐心等待（根据网速而定，可能需要数十分钟到数小时不等）。
  - `texlive` 自带 `standalone` 包和 `pdftocairo`
- `inkscape` 和 `ghostscript` 可采用默认安装或者改变一下安装路径（非 C 盘）
- 关于 `pdf2svg` ，请在 github 页面下载 zip 压缩包，然后解压出对应系统位元的版本（32 位或 64 位）。之后把解压出来的文件夹路径加到系统的环境变量 `Path` 中即可。
- 请保证所有软件的路径都加到系统环境变量 `Path` 中，否则 Windows 的 CMD 会无法找到它们（除非用完整路径）。此点会在*配置系统环境变量 Path* 详述。

2.1 配置系统环境变量 `Path`

当在 CMD 中键入非完整路径时，譬如调用 `pdftocairo` 时，只键入 “pdftocairo” 而不是它完整的安装路径时（譬如 “c:\一些文件夹\pdftocairo.exe”），系统会查找保存在 `Path` 变量中的路径，看能不能找到。故此，为了便利和兼容，一般情况下软件安装时都会把自身重要的路径加到 `Path` 中。

当然，也有列外的情况，比如不用安装的软件（譬如 `pdf2svg` ），或者用户没有修改环境变量的权限。这些情况下就需要手动把路径加到 `Path` 中。权限不足的用户需要管理员的帮助，或者进行提权。

在 Windows 上配置环境变量有好几种方法，此处描述基于 Windows 7 的一种方法。Windows 10 的方法基本一样，只不过微软把界面做了一些优化。

> **警告：** 注意，在改动环境变量时请先进行备份。

步骤如下。

1. 打开系统的控制面板
2. 点击右上角的查看方式并设为大图标
3. 点击“系统”
4. 点击“高级系统设置”
5. 点击“环境变量”
6. 选中“系统环境变量”下的 `Path`
7. 点击“编辑”
8. 在弹出的窗口中，复制所有路径并保存到用以备份用的纯文本文件
9. 在弹出窗口的路径结尾，键入分号“;”（英文的），之后粘贴入需要加入的文件夹路径（不要把文件的完整路径加进去）（Windows 10 有友好的 GUI，不需要键入分号）
10. 点击所有确认键

2.2 测试是否成功修改

假设加入的是 `pdftocairo` 的路径，那么，打开 Windows 的 CMD，键入如下命令：

```
pdftocairo --help
```

如果配置 `Path` 成功，那一系列的帮助信息将会显示在 CMD 里面。如果不成功，那 CMD 会说找不到 `pdftocairo` 。

注解： 可能需要重启电脑

你可能需要重启电脑才能令环境变量生效。若重启后仍没有生效，则证明配置错误。

一般来说，`texlive` 会自动添加路径，但 `inkscape` ， `ghostscript` 和 `pdf2svg` 都需要手动添加路径。

CHAPTER 3

极简教程

本章意在提供最简短而必要的步骤，以使用户快速上手。

在应用本章步骤前，请先保证所有需要的软件和配置已完成。

3.1 转换为 SVG

1. 把本教程附带的 `util` 文件夹复制到需要转换的 TEX 文件所在之目录下。
2. 对需要转换的 TEX 的文件的 `documentclass` 进行如下配置：

```
\documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
% 'out_svg' 是用来存放 SVG 的文件夹
% 'out_svg' 是用來存放 SVG 的文件夾
% 'out_svg' is the destination folder for SVG files
call ./util/mk_folder out_svg
&& cd /d out_svg
&& call ../util/pdf_to_svg ../\infile\space \outfile\space
}}]{standalone}
```

3. 使用 `-shell-escape` 参数对 TEX 文件进行编译。例如（需要把尖括号，及其所包裹的内容更换成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的文件名>.tex
```

4. 转换好的 SVG 文件将存放在 `out_svg` 文件夹下。

3.2 转换为 PNG

1. 把本教程附带的 util 文件夹复制到需要转换的 TEX 文件所在之目录下。
2. 对需要转换的 TEX 的文件的 documentclass 进行如下配置：

```
\documentclass[tikz, convert, convert={command=\unexpanded{
% 'out_png' 是用来存放 PNG 的文件夹
% 'out_png' 是用來存放 PNG 的文件夾
% 'out_png' is the destination folder for PNG files
call ./util/mk_folder out_png
&& cd /d out_png
&& call ../util/pdf_to_png.bat 600 ../\infile\space
}}]{standalone}
```

3. 使用 -shell-escape 参数对 TEX 文件进行编译。例如（需要把尖括号，及其所包裹的内容更换成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的文件名>.tex
```

4. 转换好的 PNG 文件将存放在 out_png 文件夹下。

3.3 转换为 EMF

1. 把本教程附带的 util 文件夹复制到需要转换的 TEX 文件所在之目录下。
2. 对需要转换的 TEX 的文件的 documentclass 进行如下配置：

```
\documentclass[tikz, convert, convert={outext=.pdf, command=\unexpanded{
% 'out_emf' 是用来存放 EMF 的文件夹
% 'out_emf' 是用來存放 EMF 的文件夾
% 'out_emf' is the destination folder for EMF files
call ./util/mk_folder out_emf
&& call ./util/gs_split_pdf.bat out_emf \outfile\space \infile\space
&& cd /d out_emf
&& call ../util/pdf_to_emf.bat
&& del /F *.pdf \sapce
}}]{standalone}
% inkscape 只能实现单张的 PDF 转换 EMF，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分页，然后调用 inkscape 做循环，把所有单页的 PDF 转换为 EMF，最后删除所
% 有单页的 PDF，只保留 EMF。
% inkscape 只能實現單張的 PDF 轉🄴 EMF，所以要先用 ghostscript 把 LaTex 生成的
```

（下页继续）

(续上页)

```
% PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉🄵爲 EMF，最後🄵除所
% 有單頁的 PDF，只保留 EMF。
% inkscape can only convert single page PDF to EMF. Therefore, the whole
% PDF generated by LaTex needs to be split into single pages first, by
% ghostscript. Then use inkscape in a loop to convert all single page
% PDFs into EMFs. Finally, delete all single page PDFs and keep only the
% EMFs.
```

3. 使用 -shell-escape 参数对 TEX 文件进行编译。例如（需要把尖括号，及其所包裹的内容更换成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的文件名>.tex
```

4. 转换好的 EMF 文件将存放在 out_emf 文件夹下。

3.4 转换为 EPS

1. 把本教程附带的 util 文件夹复制到需要转换的 TEX 文件所在之目录下。
2. 对需要转换的 TEX 的文件的 documentclass 进行如下配置：

```
\documentclass[tikz, convert, convert={outext=.pdf,
command=\unexpanded{{
% 'out_eps' 是用来存放 EPS 的文件夹
% 'out_eps' 是用來存放 EPS 的文件夾
% 'out_eps' is the destination folder for EPS files
call ./util/mk_folder out_eps
&& call ./util/gs_split_pdf.bat out_eps \outfile\space \infile\space
&& cd /d out_eps
&& call ../util/pdf_to_eps.bat
&& del /F *.pdf \sapce
}}}]{standalone}
% inkscape 只能实现单张的 PDF 转换 EPS，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分页，然后调用 inkscape 做循环，把所有单页的 PDF 转换为 EPS，最后删除所
% 有单页的 PDF，只保留 EPS。
% inkscape 只能實現單張的 PDF 轉🄵 EPS，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉🄵爲 EPS，最後🄵除所
% 有單頁的 PDF，只保留 EPS。
```

(下页继续)

(续上页)

```
% inkscape can only convert single page PDF to EPS. Therefore, the whole
% PDF generated by LaTex needs to be split into single pages first, by
% ghostscript. Then use inkscape in a loop to convert all single page
% PDFs into EPSs. Finally, delete all single page PDFs and keep only the
% EPSs.
```

3. 使用 `-shell-escape` 参数对 TEX 文件进行编译。例如（需要把尖括号，及其所包裹的内容更换成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的文件名>.tex
```

4. 转换好的 EPS 文件将存放在 `out_eps` 文件夹下。

本章中为了方便排错，命令是分部进行的（通过 `&&` 连成一行）。这些命令其实是可以放在同一个脚本中，而简化 `documentclass` 的设置的。详情请看一步到位 。

CHAPTER 4

LaTex standalone 包的配置

本教程是基于由 Martin Scharrer 开发的包（自带 standalone 类），故此于此对此包稍作讲解。

注解： standalone (*complex*)[1]

standalone 是 LaTex 中非常有用的一个包。本教程主要讲述怎样利用此包来进行图片的转换，但此包其实还有其它相当多的应用。Overleaf 上有一个非常有用的教程。

4.1 standalone 的转换命令配置

standalone 本身的 说明文档 已经对配置有详尽的说明，此处重点说一下转换成图片需要用到的 convert 选项。

配置 convert 需要在 documentclass 中进行。以下是一个利用 pdf2svg 转换为 SVG 的范例配置。

```
\documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
pdf2svg \infile\space \outfile\space all
}}]{standalone}
```

其中，

- **tikz** 此选项告诉 standalone LaTex 文档中存在 tikz 图片。
- **convert** 此选项开启 standalone 的转换功能。
- **convert={}** 此选项是 convert 的详细配置项。

[1] Ghost In Shell : Standalone Complex

- `outext=.svg` 设置输出文件的后缀名为 ".svg"。更详细的说明请参看 `standalone` 本身的 说明文档 中的表 1。
- `command=\unexpanded{}` 此项是将要调用系统运行的命令。
- `pdf2svg` 调用的转换工具。

注解： `pdf2svg` 的语法

`pdf2svg` 的语法可以参看 这里。

其中，将一多页 PDF 转换为分页的多个 SVG 的语法为：

```
pdf2svg < 输入文件名>.pdf < 输出文件名>%d.svg all
```

注意，尖括号，及其所包裹中的内容需要替换为所需的文件名。

- `\infile` 输入文件名，包含后缀名。默认后缀名为 ".pdf" 或 ".ps"。更详细的说明请参看 `standalone` 本身的 说明文档 中的表 1。
- `\space` 空格。若不使用此参数，`\infile` 后不会有空格，无论你实际上键入了多少个。`\outfile` 也是这样。
- `\outfile` 输出文件名，包含后缀名。默认后缀名为 ".png"。此处已经通过 `outext` 更改为 ".svg"。更详细的说明请参看 `standalone` 本身的 说明文档 中的表 1。

SVG 配置范例中之命令将会被翻译为如下（可以通过查看 LOG 文件确认）。其中，`mew_to_svg` 为所用的 TEX 文件的文件名。

```
1 pdf2svg mwe_to_svg.pdf mwe_to_svg-%01d.svg all
```

由此可以看出，转换的重点，是要把 `convert={}` 中的配置正确设置，以令 LaTex 将其翻译成正确的系统命令来进行图片的转换。用户可以把多个系统命令整合为一行，以做出丰富多彩的组合来达成不同的目标（在 Windows 中可以通过 "&" 或 "&&" 把多行命令合并为一行）。在转换流程 中将会详细叙述各种图片转换的流程。

注解： 运行系统命令

其实在本小结就可以看出，既然 `standalone` 可以调用以上的命令，那当然也可以调用其它系统命令。理论上，用户可以调用各种命令来做各种事，不仅仅是图片的转换。如果你有兴趣，应该可以做到编译完后自动上传到某个网络位置，或者删除整个硬盘这一类有趣的事情。

4.2 编译命令

`standalone` 需要在编译时使用 `-shell-escape` 参数。一个使用 `xelatex` 对 `mew_to_svg.tex` 进行编译的命令如下（用 `xelatex` 是因为需要处理中文）。如果你使用 LaTex 编辑器进行书写，比如 `TEXsutdio`，则需要在其中编辑其命令。你也当然可以直接在 TEX 文件所在之目录下打开 CMD，用命令直接编译。

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape mwe_to_svg.tex
```

4.3 简例

以下提供一个转换为单页多个 SVG 的简例。详细的例子会在后文说明。

以下的文件可以在此项目的根目录和 mew 文件夹中找到。

主文件：

mew 文件夹中的 mwe_to_svg.tex

```
% 这是一个将 tikz 图片转换成多张 SVG 的示例文件，使用 pdf2svg 来实现转换
% 這是一個將 tikz 圖片轉🄵成多張 SVG 的示例文件，使用 pdf2svg 來實現轉🄵
% This is a demo file for tikz to multiple SVGs using pdf2svg
\documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
    pdf2svg \infile\space \outfile\space all
}}]{standalone}

\usepackage{xeCJK}
\setCJKmainfont{Microsoft YaHei}

\usepackage{scalefnt}
\usepackage{tikz}

% tikz 和 colour 的设定
% tikz 和 colour 的設定
% tikz and colour configs
\input{../configs_tikz.tex}
\input{../configs_colour.tex}

\begin{document}

    % 全局字体缩放
    % 全局字體縮放
    % global font scale
    \scalefont{1.3}

    % tikz 图像文档
    % tikz 圖像文檔
    % tikz pics file
    \input{../tikz_pics.tex}

\end{document}
```

tikz 配置文件：

本教程根目录中的 configs_tikz.tex

```
% 以下是关于 tikz 中画流程图的设置
% 以下是關於 tikz 中畫流程圖的設置
% configure flowchart shapes
\usetikzlibrary{shapes.geometric, arrows, positioning, calc}

% start, end shape
\tikzstyle{startstop} = [rectangle, rounded corners, minimum width=3cm,
minimum height=1cm,text centered,  text=white, draw=black,
fill=colorStarstop]

% process shape
\tikzstyle{process} = [rectangle, minimum width=3cm, minimum height=1cm,
text centered, text=white, draw=black, fill=colorPro]

% decision shape
\tikzstyle{decision}=[diamond, minimum width=3cm, minimum height=1cm,
text centered, draw=black, fill=colorDec]

% comment shape
\tikzstyle{comment}=[dashed, draw=black, fill=gray!10, minimum width=3cm, minimum␣
↪height=1cm, text centered]

% docstring shape
\tikzstyle{docstring}=[draw=orange, fill=white, minimum width=50mm, text width=80mm,␣
↪minimum height=1cm]

% arrows shape
\tikzstyle{arrow} = [ultra thick,->,>=stealth, line width=1.5mm]

% comment shape
\tikzstyle{comment}=[dashed, draw=black, fill=gray!10, minimum width=3cm, minimum␣
↪height=1cm, text centered]

\tikzset{
    subprocess/.style = {rectangle, draw=black, semithick, fill=orange!30,
        minimum width=#1, minimum height=1cm, inner xsep=3mm, % <-- changed
        text width =\pgfkeysvalueof{/pgf/minimum width}-2*\pgfkeysvalueof{/pgf/inner␣
↪xsep},
        align=flush center,
        path picture={\draw
            ([xshift =2mm] \ppbb.north west) -- ([xshift= 2mm] \ppbb.south west)
```

(下页继续)

(续上页)

```
            ([xshift=-2mm] \ppbb.north east) -- ([xshift=-2mm] \ppbb.south east);
        },
    },
    subprocess/.default = 24mm % <-- added
}% end of tikzset

\usetikzlibrary{positioning}
```

color 配置文件：

本教程根目录中的 configs_colour.tex

```
% 以下是颜色的设置
% 以下是🄵色的設置
% colour defs
\usepackage{color}
\definecolor{colorStarstop}{RGB}{174, 23, 21}
\definecolor{colorPro}{RGB}{0, 175, 121}
\definecolor{colorDec}{RGB}{255, 192, 0}
\definecolor{colorYes}{RGB}{51, 153, 51}
\definecolor{colorNo}{RGB}{255, 0, 0}
```

CHAPTER 5

转换流程

本章将对转换流程进行讲解。鉴于使用到之软件，转换成 PNG 的流程与转换成 SVG 之流程相近，而转换成 EPS 之流程则于转换成 EMF 之流程同理。

本教程中转换的流程将利用到数份 Windows 的批处理脚本（在本教程的 `util` 文件夹中），本教程将在脚本详解 中对它们进行详细讲解。

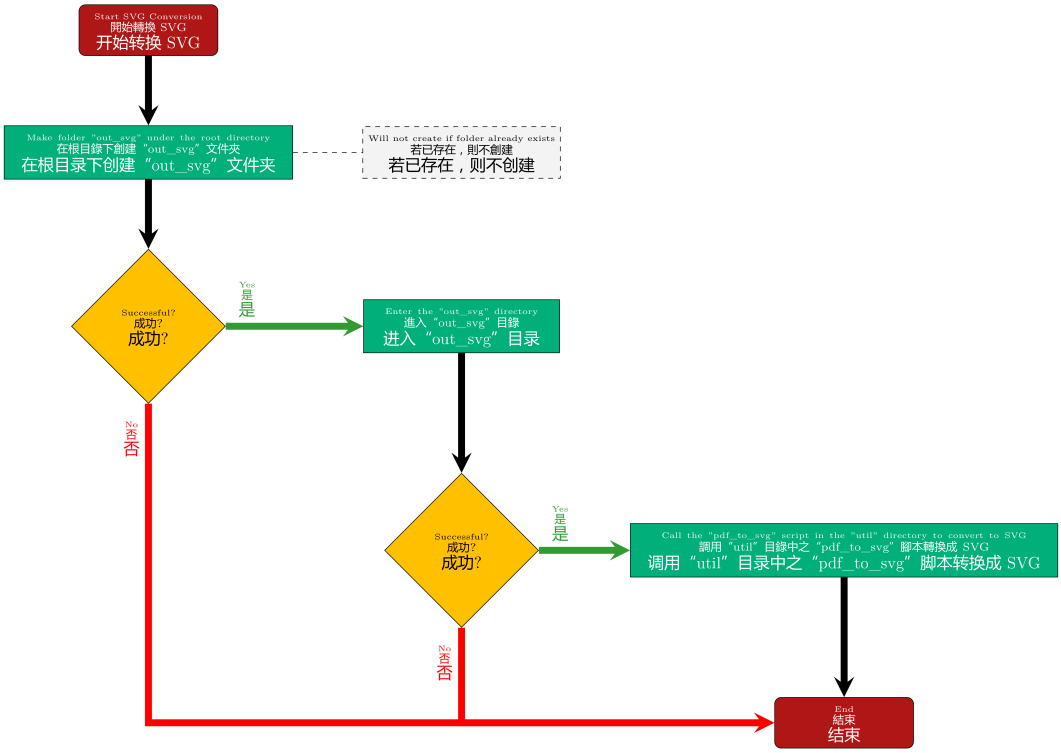
本章中之方法一律需要使用 `-shell-escape` 参数来进行编译。一个使用 `xelatex` 对 `mew_to_svg.tex` 进行编译的命令如下（用 `xelatex` 是因为需要处理中文）。

```
1 xelatex -synctex=1 -interaction=nonstopmode -shell-escape mwe_to_svg.tex
```

请把 `mwe_to_svg` 替换为你的 TEX 文件文件名。

5.1 转换为 SVG 之流程

转换为 SVG 之流程可用如下之流程图表示：



如上图所示，转换流程将在所在之目录创建一个名为 `out_svg` 的文件夹单独存放生成的 SVG 文件。之后将会调用 `util` 文件夹中的 `pdf_to_svg` 脚本来实现转换。

此流程需要用户对需要转换之 TEX 文件的 `documentclass` 进行如下配置：

```
\documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
% 'out_svg' 是用来存放 SVG 的文件夹
% 'out_svg' 是用來存放 SVG 的文件夾
% 'out_svg' is the destination folder for SVG files
call ./util/mk_folder out_svg
&& cd /d out_svg
&& call ../util/pdf_to_svg ../\infile\space \outfile\space
}}]{standalone}
```

注解： 设置 `outext`

当 `outext` 有设置时，`standalone` 会自动地在输出文件（即是 `outfile`）的文件名（不含后缀名）后加上计数关键字（一般是 `%d`）。

这小节的方法正是利用 `standalone` 之此特性，结合 `pdf2svg` 的语法来进行 PDF 转换为分页的 SVG。

> **警告：** 关于“%”和“\”符号
>
> 在 LaTex 中，“%”是一个保留字，用来表示注释。如果直接使用在 `documentclass` 之中，则

会把其后面的同行代码全部注释掉。这样的话，编译时会出问题。

然而，若用“\”进行转义（即“escape”）的话 `standalone` 是会把“\”符号作为明文加入到命令中的。这样一来，命令通常都不对，因为“\”在 LaTex 中是表示的是后面跟的是参数或者命令。而在 Windows 命令中“%”通常用来指代参数，在 Windows 中使用 for 循环时绝对会用到它，无法避免。

综上所述，如果在 `documentclass` 里面直接把系统命令写全的话，很难保证其正确性。

故此，作者选择把命令封装到多个批处理脚本中，这样就可以避免以上提及的符号问题同时方便排错。

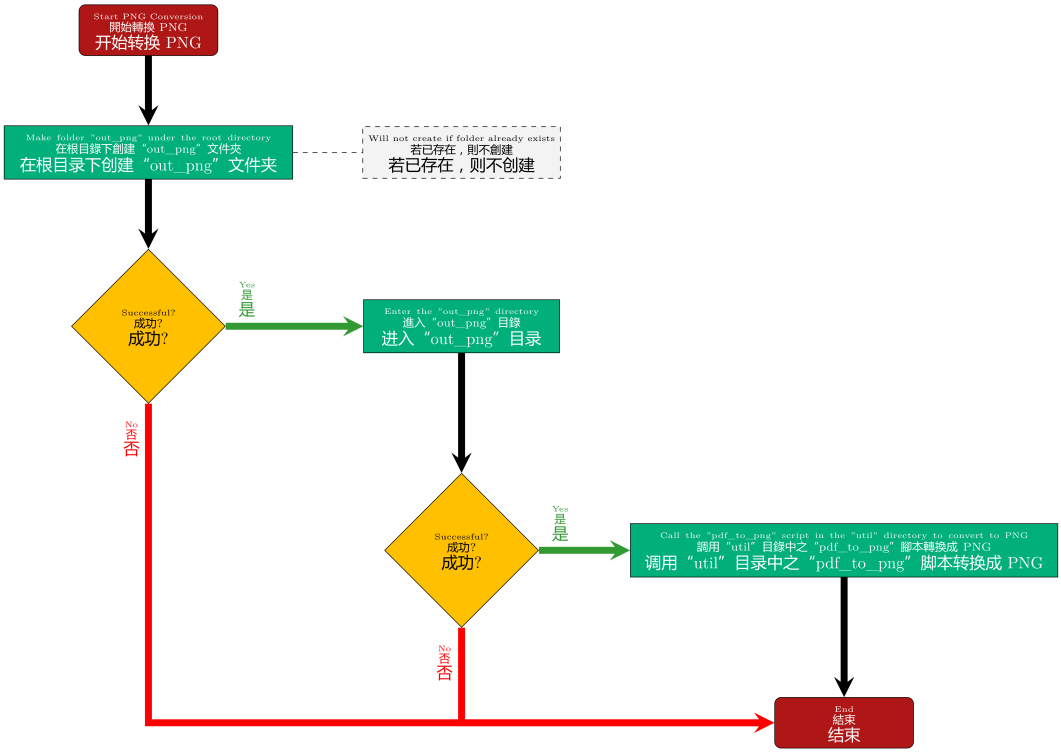
本方法用到以下两份脚本：

1. `mk_folder` 创建文件夹。
2. `pdf_to_svg` 将多页的 PDF 转换为分页的 SVG。

它们的详细讲解在脚本详解 中。

5.2 转换为 PNG 之流程

转换为 PNG 之流程可用如下之流程图表示：



如上图所示，转换流程将在所在之目录创建一个名为 `out_png` 的文件夹单独存放生成的 PNG 文件。之后将会调用 `util` 文件夹中的 `pdf_to_png` 脚本来实现转换。

此流程需要用户对需要转换之 TEX 文件的 `documentclass` 进行如下配置：

```
\documentclass[tikz, convert, convert={command=\unexpanded{
% 'out_png' 是用来存放 PNG 的文件夹
% 'out_png' 是用來存放 PNG 的文件夾
% 'out_png' is the destination folder for PNG files
call ./util/mk_folder out_png
&& cd /d out_png
&& call ../util/pdf_to_png.bat 600 ../\infile\space
}}]{standalone}
```

注解： pdftocairo

pdftocairo 会自动地把一多页的 PDF 自动分割为多张 PNG。故此只需要把输入文件给它即可（即 infile），而不需要设置输出文件（即 outfile）。

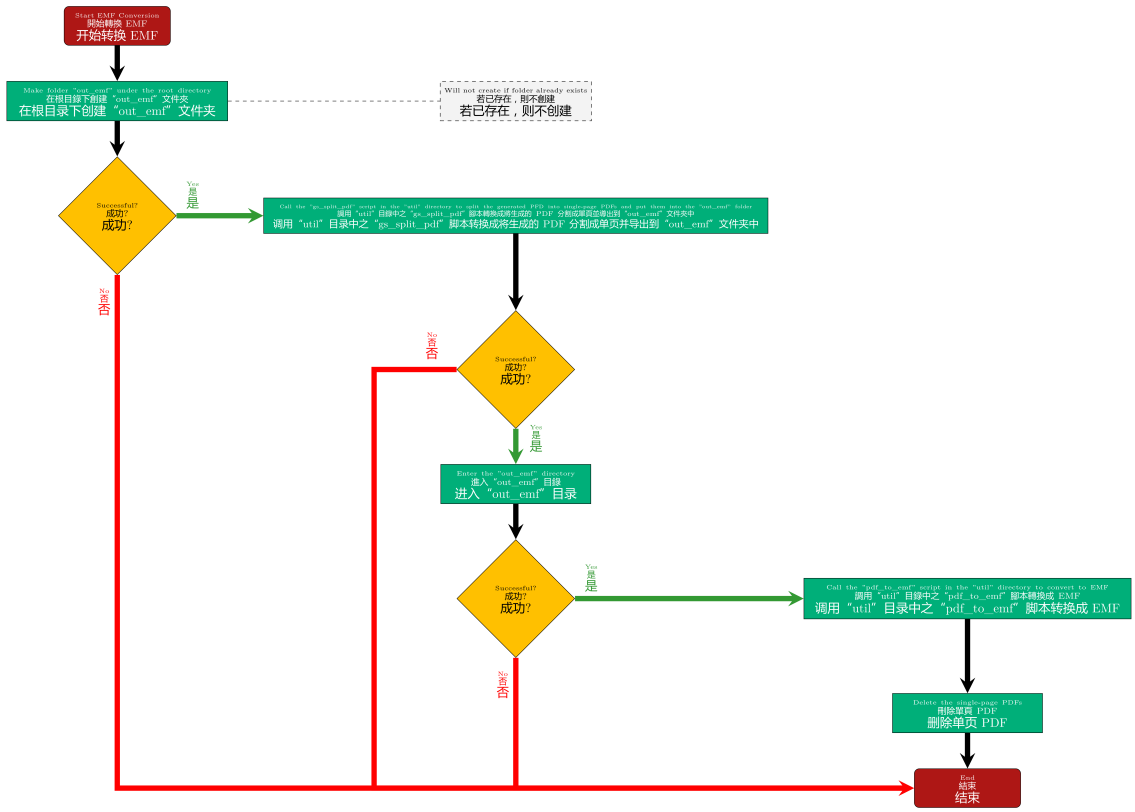
本方法用到以下两份脚本：

1. **mk_folder** 创建文件夹。
2. **pdf_to_png** 将多页的 PDF 转换为分页的 PNG。

它们的详细讲解在脚本详解 中。

5.3 转换为 EMF 之流程

转换为 EMF 之流程可用如下之流程图表示：



如上图所示，转换流程将在所在之目录创建一个名为 `out_emf` 的文件夹单独存放生成的 EMF 文件。之后将会调用 `util` 文件夹中的 `gs_split_pdf` 来对生成的 PDF 进行分页，`pdf_to_emf` 脚本来实现转换。转换完毕后会删除所有单页之 PDF，只保留 EMF。

此流程需要用户对需要转换之 TEX 文件的 `documentclass` 进行如下配置：

```
\documentclass[tikz, convert, convert={outext=.pdf, command=\unexpanded{
% 'out_emf' 是用来存放 EMF 的文件夹
% 'out_emf' 是用來存放 EMF 的文件夾
% 'out_emf' is the destination folder for EMF files
call ./util/mk_folder out_emf
&& call ./util/gs_split_pdf.bat out_emf \outfile\space \infile\space
&& cd /d out_emf
&& call ../util/pdf_to_emf.bat
&& del /F *.pdf \sapce
}}]{standalone}
% inkscape 只能实现单张的 PDF 转换 EMF，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分页，然后调用 inkscape 做循环，把所有单页的 PDF 转换为 EMF，最后删除所
% 有单页的 PDF，只保留 EMF。
% inkscape 只能實現單張的 PDF 轉🄴 EMF，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉🄴爲 EMF，最後🄴除所
% 有單頁的 PDF，只保留 EMF。
% inkscape can only convert single page PDF to EMF. Therefore, the whole
% PDF generated by LaTex needs to be split into single pages first, by
```

（下页继续）

(续上页)

```
19 % ghostscript. Then use inkscape in a loop to convert all single page
20 % PDFs into EMFs. Finally, delete all single page PDFs and keep only the
21 % EMFs.
```

注解： `inkscape`

`inkscape` 在转换时不会自动对 PDF 进行分页。若直接使用 `inkscape` 对多页 PDF 进行转换，只有第一页会被转换。

故此，本方法会先调用 `ghostscript` 对 PDF 进行分页，然后在调用 `inkscape` 对所有的单页 PDF 进行转换。

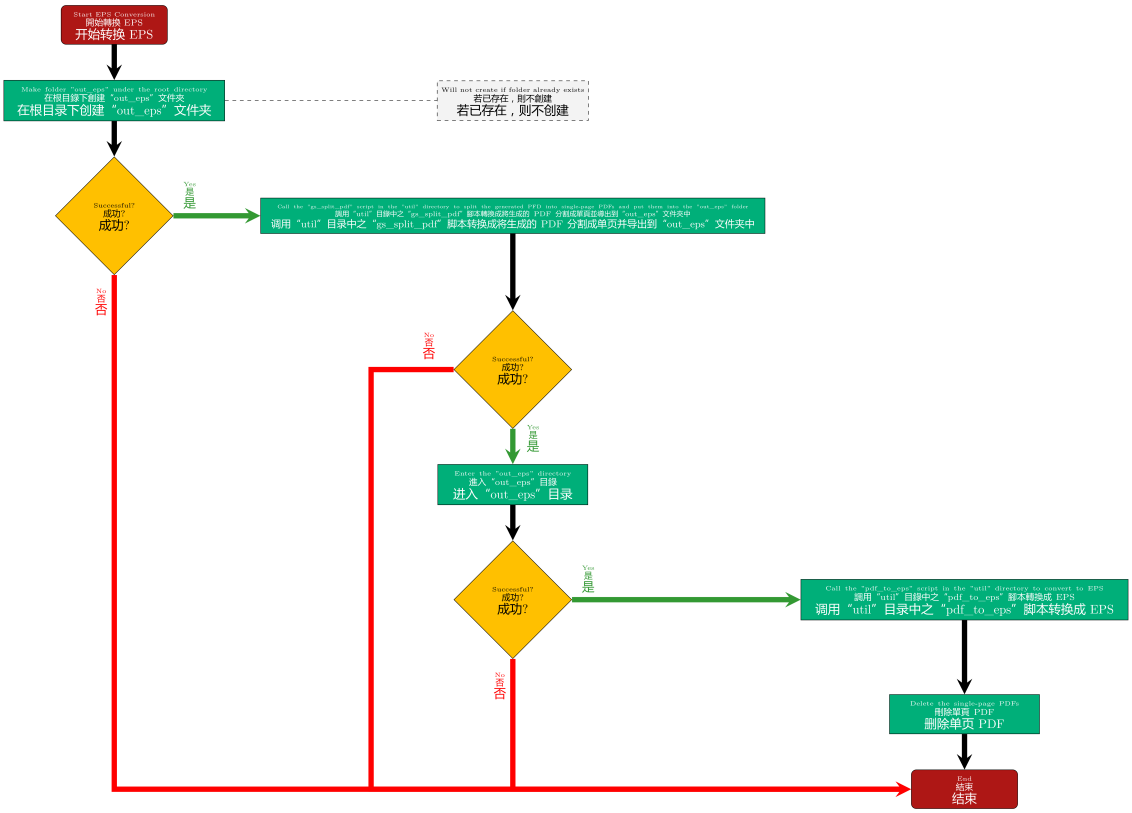
本方法用到以下三份脚本：

1. **`mk_folder`** 创建文件夹。
2. **`gs_split_pdf`** 将多页的 PDF 分割为单页的 PDF。
3. **`pdf_to_emf`** 将 PDF 转换为 EMF（仅一页）。

它们的详细讲解在脚本详解 中。

5.4 转换为 EPS 之流程

转换为 EPS 之流程可用如下之流程图表示：



如上图所示，转换流程将在所在之目录创建一个名为 out_eps 的文件夹单独存放生成的 EMF 文件。之后将会调用 util 文件夹中的 gs_split_pdf 来对生成的 PDF 进行分页，pdf_to_eps 脚本来实现转换。转换完毕后会删除所有单页之 PDF，只保留 EMF。

此流程需要用户对需要转换之 TEX 文件的 documentclass 进行如下配置：

```
\documentclass[tikz, convert, convert={outext=.pdf,
command=\unexpanded{{
% 'out_eps' 是用来存放 EPS 的文件夹
% 'out_eps' 是用來存放 EPS 的文件夾
% 'out_eps' is the destination folder for EPS files
call ./util/mk_folder out_eps
&& call ./util/gs_split_pdf.bat out_eps \outfile\space \infile\space
&& cd /d out_eps
&& call ../util/pdf_to_eps.bat
&& del /F *.pdf \sapce
}}}]{standalone}
% inkscape 只能实现单张的 PDF 转换 EPS，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分页，然后调用 inkscape 做循环，把所有单页的 PDF 转换为 EPS，最后删除所
% 有单页的 PDF，只保留 EPS。
% inkscape 只能實現單張的 PDF 轉🄴 EPS，所以要先用 ghostscript 把 LaTex 生成的
% PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉🄴爲 EPS，最後🄴除所
% 有單頁的 PDF，只保留 EPS。
% inkscape can only convert single page PDF to EPS. Therefore, the whole
% PDF generated by LaTex needs to be split into single pages first, by
% ghostscript. Then use inkscape in a loop to convert all single page
% PDFs into EPSs. Finally, delete all single page PDFs and keep only the
% EPSs.
```

本方法之原理于转换为 EMF 的流程完全一样。都是先对生成的 PDF 进行分页，再调用 inkscape 进行处理。

本方法用到以下三份脚本：

1. **mk_folder** 创建文件夹。
2. **gs_split_pdf** 将多页的 PDF 分割为单页的 PDF。
3. **pdf_to_eps** 将 PDF 转换为 EPS（仅一页）。

它们的详细讲解在脚本详解 中。

CHAPTER 6

脚本详解

本章将对本教程中所用到之脚本进行讲解。

本项目所使用到的脚本存放在 `util` 文件夹之中，它们为：

- **`mk_folder.bat`** 用于创建文件夹。

  注解： 脚本命名问题

  在 Windows 当中，如果把批处理脚本命名为所要调用的系统命令名，很可能会导致死循环。故此命名此脚本为 `mk_folder.bat` 而不是 `mkdir.bat`

- **`gs_split_pdf.bat`** 调用 `ghostscript` 把 PDF 分割为单页。
- **`pdf_to_svg.bat`** 把多页 PDF 转换为分页的 SVG。
- **`pdf_to_png.bat`** 把多页 PDF 转换为分页的 PNG。
- **`pdf_to_emf.bat`** 把 PDF 转换为 EMF（仅一页）。
- **`pdf_to_eps.bat`** 把 PDF 转换为的 EPS（仅一页）。

警告： 路径

在设置 `documentclass` 时，需要注意输入的路径参数。本教程一律使用相对路径。

`./` 是指当前所在路径。

`../` 是指往上移一层目录。

6.1 mk_folder.bat 详解

mk_folder.bat 的内容如下：

```
@echo off
REM this file is not call "mkdir" to avoid run errors

set dst_dir=%1

if not exist %dst_dir%/NUL mkdir %dst_dir%
```

此脚本用于在 CMD 当前所在之目录为根来创建文件夹。如果该文件夹已存在，则不创建而退出。

此脚本接受一个参数。此参数为将要创建的文件夹路径。

此脚本之使用方法如下：

```
mk_folder < 要创建之文件夹路径>
```

此脚本的语法和 mkdir 命令的语法一致。此处给出一个例子，假设要在目前的目录下创建一个名为 a 的文件夹，其下有一个子目录，名为 b，而 b 之下又有一个子目录，名为 c 。

```
mk_folder a\b\c
```

若文件夹路径中有空格，则需要把路径放在两个双引号之中，如：

```
mk_folder "a b\c d"
```

6.2 gs_split_pdf.bat 详解

gs_split_pdf.bat 的内容如下：

```
@echo off
REM use 64-bit ghosescript to split a pdf

set temp_folder=%1
set outfile=%2
set infile=%3

REM the gswinXXc.exe does not prompt the ghostscript window, the ones without the "c"␣
↪do prompt

gswin64c -sDEVICE=pdfwrite -dSAFER -dNOPAUSE -dBATCH -sOutputFile=%temp_folder%/
↪%outfile% %infile%
```

此脚本用于把输入的 PDF 分割为单页的 PDF。

此脚本接受三个参数。它们如下（按顺序）：

1. 输出的单页 PDF 的文件夹的路径
2. 输出的单页 PDF 文件的共用文件名加上 “%d”
3. 需要分页的 PDF 的路径

此脚本调用的是 64 位的 ghostscript ，若你安装的是 32 位的版本则需要把此脚本中的 gswin64c 更改为 gswin32c 。

此处给出一个例子，输出的文件夹为 output （已存在），共用文件名为 common，需要分页的 PDF 为 pdf_in 。

```
gs_split_pdf output common%d.pdf pdf_in.pdf
```

注解: “%d”

如果不在共用文件名后加上 “%d”，ghostscript 则不会对 PDF 进行分页。

6.3 pdf_to_svg.bat 详解

pdf_to_svg.bat 的内容如下：

```
@echo off
REM use pdf2svg (https://github.com/jalios/pdf2svg-windows) to convert a pdf to individual svgs

set inputfile=%1
set outputfile=%2

pdf2svg %inputfile% %outputfile% all
```

此脚本用于把 PDF 转换为 SVG。

此脚本接受两个参数。它们如下（按顺序）：

1. 需要转换的 PDF 文件路径
2. 输出的单页 PDF 文件的共用文件名加上 “%d”

此脚本调用 pdf2svg 来进行文件的转换。此处给出一个例子，需要转换的 PDF 名为 pdf_in，输出的单页 PDF 的共用文件名为 pdf_out ，输出的文件夹为 out_svg （已存在）。

```
pdf_to_svg pdf_in.pdf .\out_svg\pdf_out%d.svg
```

注解: “%d”

如果不在共用文件名后加上 “%d”，则只会转换 PDF 的第一页。

6.4 pdf_to_png.bat 详解

pdf_to_png.bat 的内容如下：

```
@echo off
REM use pdftocairo to convert a pdf into multiple pngs

set dpi=%1
set inputfile=%2

pdftocairo -r %dpi% -png %inputfile%
```

此脚本用于把 PDF 转换为 PNG。此脚本会在 CMD 当前所在之目录创建 PNG 文件。

此脚本接受两个参数。它们如下（按顺序）：

1. 转换的 PNG 的 DPI，一般为 300 或 600。过高的 DPI 会令转换过程冗长。一般而言，打印需要至少 300 DPI，一般 600 DPI 足以应付绝大部分情况。
2. 需要转换的 PDF 的路径

此脚本调用 pdftocairo 来进行文件的转换。此处给出一个例子，转换的 DPI 为 600 像素，需要转换的 PDF 名为 pdf_in 。

```
pdf_to_png 600 pdf_in.pdf
```

注解： pdftocairo

pdftocairo 会自动把多页 PDF 转换为单页的 PNG，非常方便。

注解： 为了方便，用户可以把 util 文件夹的路径加入到系统环境变量 Path 中，这样就可以直接调用脚本，而不需要在命令中指定 util 文件夹的路径。

CHAPTER 7

一步到位

本章在脚本详解 的基础上，讲解怎样把命令合并到一个脚本中从而简化 documentclass 的配置。

从脚本详解 中可以看出，本教程为了方便排错，而把命令分到多个脚本之中，逐个击破，但其实完全是可以把命令全部写入单个脚本之中，而实现简化 documentclass 的设置的。

本章将会提供这么做的范例。

7.1 一步转换成 SVG

本小结将使用 util 文件夹中之 qk_pdf_to_svg.bat 脚本，以 mwe 文件夹中的 qk_to_svg.tex 为例，展示如何一步转换到 SVG。

qk_to_svg.tex 中的 documentclass 的配置如下：

```
\documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
% 'out_svg' 是用来存放 SVG 的文件夹
% 'out_svg' 是用來存放 SVG 的文件夾
% 'out_svg' is the destination folder for SVG files
call ../util/qk_pdf_to_svg out_svg \infile\space \outfile\space
}}]{standalone}
```

其中，out_svg 是存放 SVG 的文件夹。

qk_pdf_svg.bat 脚本的内容如下：

```
@echo off

set dst_dir=%1
```

(下页继续)

(续上页)

```
set inputfile=%2
set outputfile=%3

if not exist %dst_dir%/NUL mkdir %dst_dir%

cd /d %dst_dir%

pdf2svg ../%inputfile% %outputfile% all
```

qk_pdf_to_svg.bat 中的命令，是把 demo_to_svg.tex 中的 documentclass 中的命令整合为一，以达到简化 documentclass 的配置的目的。

7.2 一步转换成 PNG

本小结将使用 util 文件夹中之 qk_pdf_to_png.bat 脚本，以 mwe 文件夹中的 qk_to_png.tex 为例，展示如何一步转换到 PNG 。

qk_to_png.tex 中的 documentclass 的配置如下：

```
\documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
% 'out_png' 是用来存放 PNG 的文件夹
% 'out_png' 是用來存放 PNG 的文件夾
% 'out_png' is the destination folder for PNG files
call ../util/qk_pdf_to_png out_png 600 \infile\space
}}]{standalone}
```

其中，out_png 是存放 PNG 的文件夹。

qk_pdf_to_png.bat 脚本的内容如下：

```
@echo off

set dst_dir=%1
set dpi=%2
set intputfile=%3

if not exist %dst_dir%/NUL mkdir %dst_dir%

cd /d %dst_dir%

pdftocairo -r  %dpi% -png ../%intputfile%
```

qk_pdf_to_png.bat 中的命令，是把 demo_to_png.tex 中的 documentclass 中的命令整合为一，以达到简化 documentclass 的配置的目的。

7.3 一步转换成 EMF

本小结将使用 util 文件夹中之 qk_pdf_to_emf.bat 脚本，以 mwe 文件夹中的 qk_to_emf.tex 为例，展示如何一步转换到 EMF。

qk_to_emf.tex 中的 documentclass 的配置如下：

```
\documentclass[tikz, convert, convert={outext=.pdf,
command=\unexpanded{
% 'out_emf' 是用来存放 EMF 的文件夹
% 'out_emf' 是用來存放 EMF 的文件夾
% 'out_emf' is the destination folder for EMF files
call ../util/qk_pdf_to_emf out_emf \outfile\space \infile\space
}}]{standalone}
```

其中，out_emf 是存放 EMF 的文件夹。

qk_pdf_to_emf.bat 脚本的内容如下：

```
@echo off

set dst_dir=%1
set outputfile=%2
set intputfile=%3

if not exist %dst_dir%/NUL mkdir %dst_dir%

gswin64c -sDEVICE=pdfwrite -dSAFER -dNOPAUSE -dBATCH -sOutputFile=%dst_dir%/
↪%outputfile% %intputfile%

cd /d %dst_dir%

for /r %%i in (*.pdf) do inkscape %%i -M %%~pni.emf

del /F *.pdf
```

qk_pdf_to_emf.bat 中的命令，是把 demo_to_emf.tex 中的 documentclass 中的命令整合为一，以达到简化 documentclass 的配置的目的。

7.4 一步转换成 EPS

本小结将使用 util 文件夹中之 qk_pdf_to_eps.bat 脚本，以 mwe 文件夹中的 qk_to_eps.tex 为例，展示如何一步转换到 EPS。

qk_to_eps.tex 中的 documentclass 的配置如下：

```
\documentclass[tikz, convert, convert={outext=.pdf,
command=\unexpanded{
% 'out_eps' 是用来存放 EPS 的文件夹
% 'out_eps' 是用來存放 EPS 的文件夾
% 'out_eps' is the destination folder for EPS files
call ../util/qk_pdf_to_eps out_eps \outfile\space \infile\space
}}]{standalone}
```

其中，out_eps 是存放 EPS 的文件夹。

qk_pdf_to_eps.bat 脚本的内容如下：

```
@echo off

set dst_dir=%1
set outputfile=%2
set intputfile=%3

if not exist %dst_dir%/NUL mkdir %dst_dir%

gswin64c -sDEVICE=pdfwrite -dSAFER -dNOPAUSE -dBATCH -sOutputFile=%dst_dir%/
↪%outputfile% %intputfile%

cd /d %dst_dir%

for /r %%i in (*.pdf) do inkscape %%i -E %%~pni.eps

del /F *.pdf
```

qk_pdf_to_eps.bat 中的命令，是把 demo_to_eps.tex 中的 documentclass 中的命令整合为一，以达到简化 documentclass 的配置的目的。

CHAPTER 8

总结

本章为此教程之总结。

本教程利用 Latex 中的 `standalone` 包和 Windows 命令和脚本，结合 `pdf2svg`, `pdftocairo`, `inkscape` 和 `ghostscript` 来实现编译 TEX 文件时，同时把生成的 PDF 转换成单页的图片文件。

本教程讲述来如何转换成 SVG，PNG，EMF 和 EPS 这四种类型的图片。

如果你需要转换为 JPG，请参考 EMF 的转换教程，自行实现。

本项目使用 `git` 进行版本控制。远端仓库存放在 GitHub 上。你可以在 这里 下载本项目，并提出问题。